Review of Application of Model-free Reinforcement Learning in Intelligent Decision

Wei Zhu, Sheng Yu, Haoran Chen, Zhenwen Gong

College of Information and Communication National University of Defense Technology, Wuhan, China

Abstract. With the continuous development of intelligent technologies, the traditional field of decision making is gradually evolving towards Intelligent Decision (ID), but with the increasing complexity of the environment, the explosive growth of data volume and the uncertainty of the decision making process, the difficulty of decision analysis is in-creasing. As a branch of Machine Learning, Reinforcement Learning (RL) uses Agent to train and generate rewards from the environment, ultimately resulting in intelligent models. Model-free Reinforcement Learning (MFRL) is a type of reinforcement learning in which an Agent does not need a predefined model of the environment, but inter-acts directly with the environment and learns autonomously to generate optimal strategies for model generation in complex environments. Model-free Reinforcement Learning techniques applied in the field of Intelligent Decision making can improve the efficiency and accuracy of decision making in complex environments. In this paper, we provide an overview of Model-free Reinforcement Learning in intelligent decision making and introduce the basic principles of reinforcement learning and its two branches (Model-based Reinforcement Learning and Model-free Reinforcement Learning). Various algorithms of Model-free Reinforcement Learning are analyzed and disassembled from two different functions (value-based function and policy-based function), and the characteristics, applicability range, and research results of each algorithm are derived. The typical applications of Model-free Reinforcement Learning in the field of intelligent decision making are classified and analyzed. Finally, a summary and outlook on the application of Model-free Reinforcement Learning in Intelligent Decision making are presented.

Keywords: artificial intelligence; model-free reinforcement learning; intelligent decision.

1. Introduction

With the deep application of artificial intelligence, intelligent decision making techniques are widely used in the fields of intelligent driving, adversarial gaming, and network planning. The decision model is the core of intelligent decision making, and the decision model of intelligent decision making can be trained and tested by AI technology on the basis of human a priori knowledge, so that it can autonomously complete the decision task in the decision space [1]. However, with the increasing amount of data and the changing environment, human a priori knowledge is no longer sufficient to meet the needs of AI technology training, for example, traditional decision models are often unable to make quick judgments in the face of unstructured environments that have never existed before. In order to generalize decision models to new environments, they need to be allowed to continuously interact with the environment to learn and improve their ability to cope with complex novel environments.

RL technology, another development in artificial intelligence, has matured with the intensive research and application of statistics, big data technology, and optimization theory, and RL trains the Agent by interacting with the environment to obtain the final model [2].

RL is different from supervised learning and unsupervised learning in machine learning. Supervised learning requires labeled data sets for training, and the trained models need to be tested with these data, so these data sets actually serve as both training and testing sets. In RL, these datasets do not exist, and the agent is trained by interacting with the environment [3], and tested by interacting with the environment, unlike supervised learning where the labeled datasets can directly judge the model, the judgment signal in RL is obtained by interacting with the environment to obtain the reward The judgment signal in RL is obtained by interacting with the environment to obtain a revised model.

Unsupervised learning, like supervised learning, also requires a dataset for support, but the difference is that this dataset is not labeled, so its learning goal is to find the hidden relationships in the data [4] and to label these data. the learning goal of RL is to maximize the gain at each step, i.e., to maximize the cumulative gain.

The differences between supervised learning, unsupervised learning and reinforcement learning are shown specifically in Table 1.

Table 1. Difference between supervised learning, unsupervised learning and reinforcement learning

	Whether the data set is available	Whether the data is labeled	Interaction objects	Learning Objectives
Supervised Learning	Yes	Yes	Data with labels	Input to output mapping relationship
Unsupervised Learning	Yes	No	Data without labels	Hidden relationships in data
Reinforcement Learning	No	No	Environment	Maximize cumulative earnings

In RL, because there are many types of environments, those that can model the environment are called modeled reinforcement learning, while those that cannot predefine the environment and can only interact with the environment through the Agent alone are called model-free reinforcement learning, as shown in Figure 1.

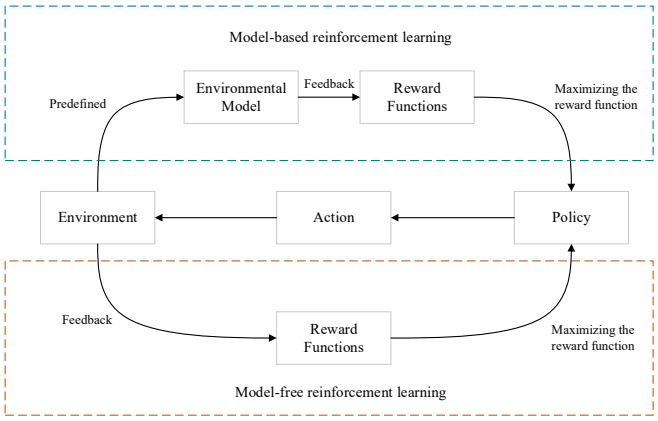

Figure 1. Sketch of two types of reinforcement learning models.

For modeled reinforcement learning, the environment needs to be modeled in advance, while in model-free reinforcement learning, it is directly interacted with the environment, and the feedback obtained from the interaction is directly input into the reward function to derive the reward value of the Agent and update the policy in real time to better perform the next action [5]. This processing, can be well applied with some complex environments, which are not directly modelable, and forcing environmental preprocessing can make the error rate of the trained Agent high.

For the field of intelligent decision making, the diversity of decision trees and the complexity of the decision environment make the "environment" itself extremely difficult to pre-process, and using modeled reinforcement learning methods to solve the problem will make the training too slow and difficult to achieve the expected results [6], so the application of model-free reinforcement learning to the field of intelligent decision making can be a good solution to the problem of training your efficiency and accuracy.

In this paper, we will review the application of model-free reinforcement learning in intelligent decision making, firstly introduce the basic principles of RL, then summarize the classical algorithms and their respective characteristics in model-free reinforcement learning, then summarize its application in intelligent decision making, and finally summarize the problems of model-free

reinforcement learning in the field of intelligent decision making and outlook the future development trend.

2. Markov Decision Process

The Agent is the entity trained in RL, and in a cycle, the Agent generates feedback by interacting with the environment, which is fed into the reward function to adjust the policy [7], which in turn influences the next action, which is a trial-and-error process, and this behavior of reaching the goal by a step-by-step cyclic trial-and-error approach is generally called sequential decision making. Markov Decision Process (MDP) is the classical paradigm to solve this problem, which describes a learning process that interacts with the environment and eventually reaches the maximized benefit.

MDP has two expressions in academia, one is quadratic (S, A, P, R)and the other is quintuple (S, A, P, R, γ). No matter how the grouping is done, S, A, P, R is always appear as the core, where S represents the state, A represents the action, P represents the state transfer probability, R represents the reward function, and γ in the quintuple represents the discount factor. The general MDP is shown in Figure 2.

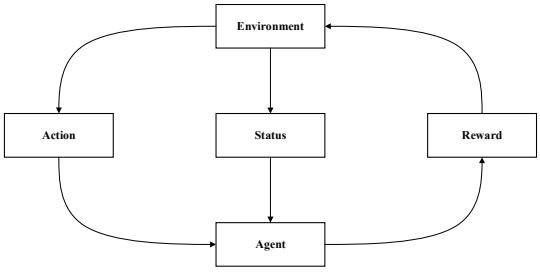

Figure 2. MDP diagram.

ww.DeepL.com/T

First make an assumption that the action executed in the current state in the MDP is only related to the state of the previous step as well as the action, and is independent of the environment as well as the Agent itself. Then according to MDPness, the probability $P_{ss'}^{a}$ that the state s of the environment is transferred to s' by the action a and the expected reward R_s^a can be expressed as:

$$P_{ss'}^{a} = P(S_{t+1} = s' | S_t = s, A_t = a) \tag{1}$$

$$R_s^a = E(R_{t+1} | S_t = s, A_t = a) \tag{2}$$

where E is the desired reward function.

The MDP reflects the Agent's actions within a cycle, but in the overall RL, the Agent's goal is to maximize the overall reward R, i.e., to maximize the expected value of the cumulative gain [8]. In Eq. (3), $v_\pi(s)$ is the state value function under the current policy π and the environment state s, which can be expressed as:

$$v_\pi(s) = E_\pi(G_t | S_t = s) \tag{3}$$

where G_t denotes the sum of all gains decaying in a certain proportion during the process from the beginning of the current state to the end of the termination state, and the decay factor is γ. Thus, $v_\pi(s)$ can in turn be expressed as:

$$\begin{aligned} v_\pi(s) &= E_\pi(R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots | S_t = s) \\ &= E_\pi(R_{t+1} + \gamma G_{t+1} | S_t = s) \\ &= E_\pi(R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s) \end{aligned} \tag{4}$$

Eq. (4) is called the Bellman equation for v_π.

RL can be regarded as a collection of multiple MDP problems, so it is necessary to find the overall optimal value function, which determines the optimal action value function and finally the optimal policy [9]. So the relationship between the optimal value function v_*, the optimal action value function q_* and the optimal strategy π_* is shown in Eqs. (5) and (6) as follows.

$$\begin{aligned} v_*(s) &= \max_a q_{\pi_*}(s,a) \\ &= \max_a E_{\pi_*}(R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a) \\ &= \max_a E\left(R_{t+1} + \gamma v_{\pi_*}(S_{t+1}) \middle| S_t = s, A_t = a\right) \\ &= \max_a R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_*(s') \end{aligned} \tag{5}$$

$$\begin{aligned} q_*(s,a) &= E(R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \,|S_t = s, A_t = a) \\ &= R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \max_{a'} q_*(s', a') \end{aligned} \tag{6}$$

3. Model-free Reinforcement Learning

For intelligent decision problems, with the continuous expansion of data and the exploration of emerging fields, the environment is often difficult to know directly using models [10], and model-free reinforcement learning works well without the advantage of using environmental models, so it is used in many areas of intelligent decision making. Model-free reinforcement learning is divided into two categories: value-based and policy-based, i.e., value-based functions and policy-based functions. In this section, we take these two categories as the core and describe each algorithm involved, and summarize the characteristics and application scope of each algorithm.

3.1 Temporal-Difference of value-based

The Temporal-Difference (TD) method combines the characteristics of dynamics in dynamic planning and the advantages of Monte Carlo algorithm to directly put the Agent into the environment for interactive learning, and during the whole RL process, the Agent does not need to wait until the end of the whole process to learn, but can learn in the MDP sequence, which are typical of model-free ideas. TD can be divided into TD(0), n-step TD, and TD(λ) [11].

TD (0) judges the current value function by the next state, where the value function of the next state is particularly critical, and the Agent uses it to explore a state backward and compare it with the present state to optimize and get the next optimal state, and the exploration process is shown in Figure 3.

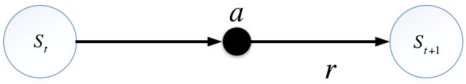

Figure 3. TD (0) flow chart.

The n-step TD is different from TD(0) in that it requires multiple steps, i.e., a series of multiple TD(0), and the Agent needs to perform the second TD(0) immediately after using the first TD(0) to get the next state, thus performing a cyclic operation, and after exploring n TD(0), the current value function and the first n-1 value cumulative rewards are used to calculate the final value function. n-step TD exploration process is shown in Figure 4.

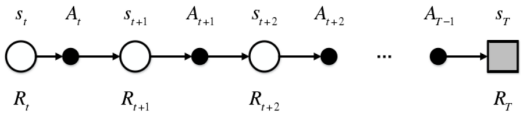

Figure 4. n-step TD flow chart.

TD (λ) differs from n-step TD in that it is not a simple series of TD (0), but a weighting of each different MDP decision chain based on n-step TD, each of which has its corresponding weight [12], and when the length of n-step TD is longer, its weight in TD (λ) is higher and vice versa, and the process of TD (λ) exploration is shown in Figure 5.

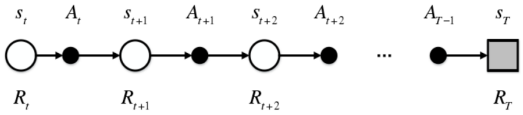

Figure 5. TD(λ) flow chart.

TD does not require an environmental model, i.e., a model that describes the joint probability distribution of the payoffs and the next step state, and naturally employs an online, fully incremental approach to implementation compared to the Monte Carlo method, which must wait until the end of a step because only then is the exact payoff value known, whereas TD simply waits until the next moment.

3.2 Deep Q-Learning Network of value-based

The Q-learning algorithm uses a Q-table to record the action values in each state, and when the state space or action space is large, the storage space required will also be large. The algorithm cannot be used if the state space or action space is continuous [13]. The Q-learning algorithm can only be used to solve discrete low-dimensional state space and action space class problems. Therefore, researchers proposed Deep Q-Learning Network (DQN) algorithm based on Q-Learning algorithm combined with deep neural network.

The core of the DQN algorithm is to replace the Q form, i.e., action value function, with an artificial neural network. The input of the network is the state information and the output is the value of each action, so the DQN algorithm can be used to solve both continuous state space and discrete action space problems.

DQN is approximated by a neural network to obtain the value function, specifically the input of the neural network is an observation (i.e., state,s) and the output is the value function Q(s,a) (a is action a). The value function is obtained through the neural network and the DQN uses an ϵ-greedy strategy to output the action. the steps are that the environment first gives an obs, the intelligence finds all the value functions Q(s,a) about this obs according to the neural network, then selects the action according to the ϵ-greedy strategy and makes This is a step. at this point we update the parameters of the value function network according to R [14]. This cycle continues until we have trained a good value function network. The algorithm flow is shown in Figure 6 below.

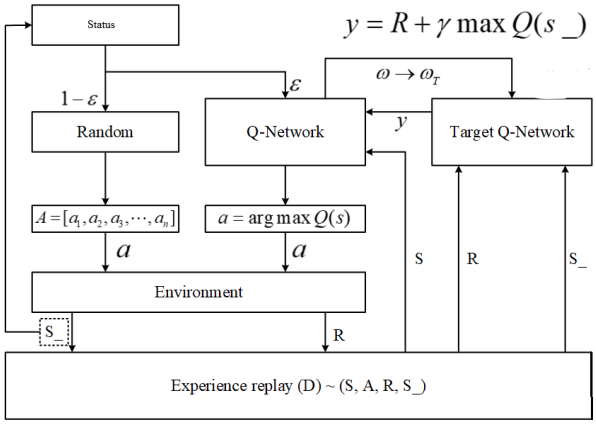

Figure 6. DQN flow char.

3.3 Actor-Critic of policy-based

The actor-critic algorithm (Actor-Critic) consists of two parts, Actor and Critic, respectively.

The value-based approach needs to show the value that can be obtained for each action corresponding to each state of the learning environment, and in a continuous environment, the number of actions is uncontrollable, which leads to the fact that the number of their values is also difficult to estimate, and the traversal approach is already undesirable [16]. Because the policy-based approach

is able to determine the next action directly based on the state of the environment, its more applicable to the case of continuous actions. The Actor-Critic algorithm, on the other hand, learns both the policy function and the value function together and uses both methods to make a decision at the same time to obtain an optimal behavior through their interaction. Actor is able to learn the corresponding action based on the state of the environment and is a Critic is a value-based approach that evaluates the value of an action, and the result of this evaluation is used as a reward for the model. This can accelerate the learning process of the strategy learning algorithm and achieve better results.

3.4 Asynchronous Advantage Actor-critic of policy-based

Since the dataset for deep reinforcement learning algorithms comes from the environment, the correlation between different data is inevitably too strong when collecting the environment. When using the DQN algorithm, an empirical pooling approach is used to reduce the problem of excessive data correlation. the A3C algorithm differs from this by using multiple threads for training to solve this problem. Each threaded sub-network is trained in the environment and updates itself as it computes the gradient of its respective loss function. The updated global network distributes a better experience to each thread simultaneously.

4. Applications in Intelligent Decision Making

4.1 Model-free reinforcement learning games for AI applications

Given the black-box nature of the learning process in high-dimensional situation spaces and pose spaces, model-free deep reinforcement of learning knowledge indicates a great test. Recently, the occurrence of DeepMind AlphaGo has given new ideas to deal with these difficulties [15]. After continued deep research, real-time strategy (RTS) games have become a hot topic, and projects such as Star AI Alpha Star and OpenAl Five have long caused high interest in AI communities around the world. In China, Tencent's official AI lab is also working on AI+ game research using model-free authorized learning.

In contrast to Go, the difficulties of RTS games are reflected in four key levels: high measurement complexity, diverse intelligence, unsound information, low rewards, and low latency low time. For the Agent to grasp the RTS game, the RL model must have the ability to measure both the actual operation of the macro countermeasures and the external economic implementation. Unbeknownst to us the key focus of recent research is on external economic implementation, which does not provide a complete solution to macro strategic control.

Wu et al. of the official Tencent AI Lab propose a novel learning-based hierarchical macro countermeasures (HMS) model to capture multiplayer online arena (MOBA) games in RTS games. Practicing the HMS model allows the Agent to make established macro-strategic decisions and further control the external economic implementation [16]. The process of macro countermeasure surgery treatment can be divided into three parts: link identification, focused predictive analysis, and implementation. Thus, Wu et al. proposed a two-level macro countermeasure architecture, containing a process level and a focus level, to model the process. The step level is used to specify the current Agent's play video level, the attention level is used to predict the best part of the hero on the terrain map for analysis, and the stage and attention levels give general specific guidance for macro implementation.

In MOBA games, multi-intelligent 1v1 scenarios involve higher spatial complexity of states and actions than traditional 1v1 scenarios, resulting in more difficulty in exploring control strategies to reach human level [17]. Ye et al. in Tencent's A1 Lab propose to use model-free deep reinforcement learning to predict Agent's game behavior, design the DRL framework and optimize the algorithm. The entire framework enables efficient exploration at scale through a low-coupling and highly scalable module design.

In MOBA games, multi-intelligent 1vs1 scenarios have higher situation and pose space complexity than traditional-style 1vs1 scenarios, causing control countermeasure exploration to be more difficult

to achieve people level. Ye et al. of Tencent's official A1 lab proposed to apply model-free deep reinforcement learning to predict and analyze the game behavior of intelligent bodies, and designed the DRL architecture to enhance the algorithm. All the architectures have low coupling and highly scalable modular design design, which can accomplish large-scale and efficient probing.

4.2 Model-free reinforcement learning games for AI applications

Designing motion control for legged intelligent robots is one of the bigger tests in the field of intelligent robot control. Traditional-style approaches, which are generally pipelined, generally must carry out very accurate models of device human dynamics [18], but designing dynamics models must be very much a matter of expertise, and this area has been one of the larger impediments to the development of intelligent robots. Model-free deep reinforcement learning gives a new orientation to deal with these challenges, and research workers have explored the use of model-free reinforcement learning in the control of intelligent robots that do not have to define a model of dynamics in advance, gaining longevity in its use level.

An efficient model-free sampling algorithm for deep gain learning and an asynchronous intelligent robot reinforcement learning system based on large entropy gain learning have been proposed by Haarnoja et al. In this work, we not only allow intelligent robots to change tasks without much tuning, but also greatly improve the retrieval efficiency of neural networks [19].

The RL system consists of three key components: a module that collects empirical data from relevant intelligent robots, a module that measures rewards based on intelligent robot parts obtained from their behavior and measurements, and a module that upgrades the neural network. The different modules operate asynchronously with each other and apply timestamps to the future data streams. During the learning process, the observation of each control process is collected and granted to the neural network to obtain the logical inference results and implement the pose to get the reward. The entire process is stored as a tuple in the cache and continues iteratively to learn the best practical operation. Finally, we can test it on a real quadrupedal intelligent robot and get excellent properties.

Applying model-free reinforcement learning to intelligent robot control is one of the key ways to handle today's complex intelligent robot control. DeepMind's Hafner et al. propose a model-free reinforcement learning architecture for legged intelligent robots to learn some complex motor behaviors. The full architecture is based on a multi-task The full architecture is based on a multi-task RL algorithm that can properly process data, apply offline countermeasures, and maintain the super-major parameters and reward settings unchanged throughout the experiment [20].

The model-free enhanced learning architecture differentiates different motion behaviors based on its own gyroscope and smart robot rate, and sets different rewards for different motion behaviors. different rewards. Different types of intelligent robots do not have the same behavior space and viewing space. For example, a three-legged intelligent robot has 9 pose For example, a three-legged robot has 9 posture rooms and 127 viewing rooms, and a six-legged robot has 18 posture rooms and 282 viewing rooms each. By making the pose space and viewing space different, the We make it mandatory for the Agent to navigate through different situation spaces during task transitions, which greatly improves the retrieval efficiency, and at the same time greatly improves the robustness of transitions during task transitions. The robustness of the transformation during task transformation is also greatly improved. Finally, experiments conducted on nine different intelligent robots applying the same RL algorithm illustrate that the architecture enables intelligent robots can learn consistently over general motor behavior without incorporating special platforms or additional learning tools.

5. Result

The problem of designing reward mechanisms is another important part of pattern-free empowerment learning. If an effective reward mechanism can be designed during the training process, then the number of interactions between the Agent and the environment can be reduced, the sample utilization can be improved, and the cost of experiments can be reduced. However, the problem of

designing reward mechanisms is subject to different experimental environments. Based on the idea of intuitive solutions, many algorithms usually use manually designed rewards, but they are not universal and easily fall into local optimal solutions.

Meanwhile, model-free reinforcement learning has great research potential for empirical reproduction, multi-objective learning, and assistance tasks. Better application of existing intelligent decision results to achieve accurate value function prediction and optimal governance will be a strong motivation for breakthroughs in intelligent decision making.

Acknowledgment

Thanks to all the partners in the laboratory for their help, and thank my family for their great support.

References

[1] Qin, W., Li, N., Liu, X., Liu, X. Lei, Tong, Q., Liu, X. Hong. A review of model-free reinforcement learning research.[J]. Computer Science,2021,48(03):180-187.

[2] Cao Lichun. Reinforcement learning target detection based on U-shaped network[D]. Inner Mongolia Normal University, 2021. doi:10.27230/d.cnki.gnmsu.2021.000747.

[3] Fei Zhu, Yang-Yang Ge, Xing-Hong Ling, Quan Liu. A model-free secure reinforcement learning method based on restricted MDP[J]. Journal of Soft-ware,2022,33(08):3086-3102.DOI:10.13328/j.cnki.jos.006318.

[4] Liu Q, Zhai J. W., Zhang Z. C., Zhong S., Zhou Q., Zhang P., Xu J.. A review of deep reinforcement learning[J]. Journal of Computer Science,2018,41(01):1-27.

[5] Lai J, Wei JINGYI, Chen XILANG. A review of hierar-chical reinforcement learning[J]. Computer Engineering and Applications,2021,57(03):72-79.

[6] He L, Shen L, Li F, Wang Z, Tang WQ. Policy reuse in re-inforcement learning: research progress[J]. Systems En-gineering and Electronics Technolo-gy,2022,44(03):884-899.

[7] Dosovitskiy A, Koltun V. Learning to act by predicting the future[DB/OL]. (2017-02-14) [2021-04-02]. hops://arxiv.org/abs/611.01779.

[8] Oh J, Chockalingam V, Singh S, et al. Control of memory, ac-tiveperception, and action in minecraft[DB/OL]. (2016-OS-30)[2021-04-02].hops://arxiv.org/abs/1605.09128.

[9] Kempka M, Wydmuch M, Runc G, et al. ViZDoom: A Doom-based AI research platform for visual reinforcement learning[C]//IEEE Conference on Computational In-telligence and Games. Piscataway, USA: IEEE, 2016: 1-8.

[10] Schulman J, Wolski F, Dhariwal P, et al. Proximal policy-optimization algorithms[DB/OL]. (2017-08-28) [2021-04-02]. hops://arxiv.org/abs/1707.06347.

[11] Levine S, Abbeel P. Learning neural network policies with guided policy search under unknown dynam-ics[C]//Advancesin Neural Information Processing Sys-tems. La Jolla, USA:Neural Information Processing Sys-tems Foundation, 2014:1071-1079.

[12] Levine S, Finn C, Darrell T, et al. End-to-end training of deep visuomotor policies[J]. The Journal of Machine Learning Re-search, 2016, 17(1): 1334-1373.

[13] van Hasselt H, Guez A, Silver D. Deep reinforcement learning with double Q-learning[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2016, 30(1): 2094-2100.

[14] Bellemare M G, Dabney W, Munos R. A distributional perspective on reinforcement learning[J]. Proceedings of Machine Learning Research, 2017, 70: 449-458.

[15] Fujimoto S, van Hoof H, Meger D. Addressing function approximation error in actor-critic methods[DB/OL]. (2018-10-22) [2021-04-02]. hops://arxiv.org/abs/1802.09477.

[16] Luo Y, Xu H, Li Y, et al. Algorithmic framework for model based deep reinforcement learning with theoretical guarantees[DB/OL]. (2021-02-15) [2021-04-02]. hops://arxiv.org/abs/1807.03858.

[17] Weber T, Racaniere S, Reichert D P, et al. Imagina-tion-augmented agents for deep reinforcement learning[DB/OL].(2018-02-14) [2021-04-02]. hops://arxiv.org/abs/1707.06203.

[18] Khansari M, Kappler D, Luo J L, et al. Action image rep-resentation: Learning scalable deep grasping policies with zero real world data[C]//IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2020: 3597-3603.

[19] Popov I, Heess N, Lillicrap T, et al. Data-efficient deep reinforcement learning for dexterous manipulation[DB/OL].(2017-04-10) [2021-04-02]. hops://arxiv.org/abs/1704.03073.

[20] Gupta A, Eppner C, Levine S, et al. Learning dexterous ma-nipulation for a soft robotic hand from human demonstra-lions[C]//IEEE/RSJ International Conference on IntelligentRobots and Systems. Piscataway, USA: IEEE, 2016: 3786-3793.